1 **COMPUTER USER INTERFACE ARCHITECTURE WHEREIN BOTH**
2 **CONTENT AND USER INTERFACE ARE COMPOSED OF DOCUMENTS WITH**
3 **LINKS**
4
5 **TECHNICAL FIELD**

6 This invention relates generally to computer user interface architectures. More

7 particularly, the invention provides a user interface architecture in which both user content

8 and user interface are composed of document pages with links.

9 **BACKGROUND OF THE INVENTION**

10 Many personal computer users find the desktop metaphor of prior art computer user

11 interfaces ("UIs") confusing and difficult to learn. Accordingly, there is a need for a

12 system that simplifies the user's interaction with the computer by using fewer kinds of user

13 interface controls in a more general way.

14 Further, the ways in which users interact with information about prior UIs is

15 different than the way the user interacts with content, such as documents, presentations,

16 and the like. For example, in prior art UIs, content and UI information are displayed

17 entirely differently. Content is typically displayed in a particular region or frame of the

18 display. User interface information is never displayed there. Instead, user interface

19 information is displayed in dialog boxes, drop down menus, and tool bars. User content

20 never shows up in dialog boxes, drop down menus, and tool bars. Similarly, users find

21 user content documents and UI help information differently. Accordingly, there is a need

22 for a UI architecture in which the concepts and actions the user must learn are the same for

23 interacting with both content and the UI. Such a unification makes computer software

24 easier and more efficient to use.

25 Prior art UIs for desktop computers typically require a keyboard and mouse in

26 order for a user to interact with them, and most pen-enabled palmtop computers have

27 cumbersome means of interaction. Therefore, there is a need for more "natural" styles of

28 interacting with a computer by using a minimum number common gestures such as touch,

29 hold, erase, draw or write.

30 Prior art UI desktop metaphors applied to small form factor devices are typically

31 cluttered and difficult to use. In addition, applications that provide rich functionality are

1 sometimes constrained by the limited ability of a user to navigate menus and dialogs of

2 prior art UIs. For instance, for such applications, the menus and tool bars may get too big,

3 and the help system may get too cumbersome to navigate or search. Accordingly, in

4 addition to the need for a simpler more unified experience for the user of an application,

5 there is also a need to facilitate the uncluttered presentation of user interfaces for

6 applications providing very rich functionality.

7 Usability data for prior art UIs show that users of multi-windowed systems don't

8 always know which actions will produce results in which window. Therefore, there is a

9 need to reduce the complexity and confusion sometimes caused by multi-windowed user

10 interfaces.

11 Prior art UIs typically offer limited capabilities for customizing the UI.

12 Accordingly, there is a need for a UI architecture that provides greater flexibility to users,

13 content developers, and third-party software developers by providing broader capabilities

14 for easily customizing the UI. For example, different groups of users may be of different

15 levels of computer skill and have need of different sets of features, and the UI can be

16 customized to better suit their needs.

17 Users of prior art UIs sometimes become extremely frustrated when their work is

18 lost because their work was not properly saved. Accordingly, there is a need to provide a

19 save-less model, so that users do not need to explicitly save their work.

20 Users of prior art UIs typically do not have a convenient and seamless way to

21 record notes verbally and to associate notes with particular parts of a document.

22 Accordingly, there is a need to provide rich support for audio note taking with the ability to

23 correlate and synchronize audio and textual material and to review and retrieve audio

24 notes.

25 Prior art device-to-device and device-to-PC synchronization schemes typically are

26 not seamless and require a great deal of configuration and attention from the user.

27 Accordingly, there is a need to provide automatic and transparent synchronization between

28 a user's computers, such as a handheld computer and a desktop computer.

29 In prior art UIs, methods for getting help are currently separate from the content

30 and often require completely different interactions than interacting with content.

2

1   Accordingly, there is a need to make the process of getting help about a function the same

2   as the process for carrying out the function.

3   Prior art UIs typically have a "single-user model" at the heart of their interface

4   metaphors, which makes sharing content and annotations with other users difficult and

5   non-intuitive. Accordingly, there is a need to make sharing and collaborating on

6   documents easier and more automatic.

7   **SUMMARY OF THE INVENTION**

8   According to various preferred embodiments, the invention includes a user

9   interface architecture in which user content and user interface are composed of documents

10   with links. A link can relate a spot or region in a document with a spot or region in another

11   document, so that touching the link causes the display to navigate to that other document.

12   A link can also relate a spot or region in a document and an active runable object such that

13   when a user activates that link or touches that spot in the document, the associated object is

14   run. Parameters for the execution of the object may be supplied by properties associated

15   with the link. Links, therefore, can act as commands. Links may be to any kind of

16   command code.

17   A link may manifest in various ways based on the link's properties. Links can look

18   like not only clickable spots, but also fill-in fields and other kinds of well-known and later-

19   developed user interface elements. Alternatively, links can manifest in the containing

20   document a frame displaying the contents of part, or all, of the linked-to document. When

21   links are displayed, in addition to basing the display format of the link on the link's display

22   properties, the link display format may depend upon the characteristics of the linked-to

23   document.

24   The path a user takes to reach a document typically affects the behavior and

25   presentation of the document. State-like information for displaying a linked-to document

26   page is stored separately from the linked-from and linked-to documents as part of the

27   link's properties. Users access, interact with, and navigate among both user content

28   documents and user interface documents in a unified way, namely, by activating links.

29   Further, both user content document pages and user interface document pages are

30   displayed in a single viewing frame. This unified approach simplifies the user's

1 interaction with both user content and user interface documents by reducing the number of

2 concepts a user must learn in order to access, interact with, and modify both user content

3 documents and the user interface.

4   A non-linear navigation history is maintained such that a user can navigate along a

5 first path, back up using a previous link an appropriate number of times, navigate along a

6 second path, back up along the second path using the previous link an appropriate number

7 of times, and re-navigate along the first path again using a next link. Every document page

8 to which a user navigates is saved in the user's navigation history. Users can query their

9 navigation histories and view their navigation history in various ways, such as: by time, by

10 appearance, by site, document, section, page, and the like. User can also view their

11 navigation history as nodes with side tracking branches, as a linear list, or as a combination

12 of most recently viewed pages and last few task categories. According to a preferred

13 embodiment, navigation from user content pages through user interface pages that results

14 in a command being executed is automatically removed from the user's view of the

15 navigational history in order to keep the navigational history view uncluttered.

16   A flexible selection model is supported allowing users to select the object of a

17 command either before or after the command itself is selected. This flexible selection

18 model allows UIs built according to the principles of this invention to scale to small

19 display areas. UIs built according to the principles of this invention also scale well to

20 applications having voluminous and/or complicated user interfaces by facilitating an

21 organized and uncluttered view of the user interface command hierarchy and/or user

22 interface help information for such applications.

23   Users get at commands by navigating to a page where the desired command is

24 found. In a preferred embodiment, the documents-with-links UI according to the

25 principles of this invention is organized to make frequently used commands a single

26 navigation step away, or through customizations, no steps away. A self-explanatory

27 document, the Guide Book is provided. The Guide Book is a readable manual that users

28 can go through in a logical order, a page at a time, like any conventional user manual.

29 Each command mention, however, is an active command instance that can be invoked in

30 place.

1   Other features and advantages of the invention will become apparent through the

2 following description, the figures, and the appended claims.

3

4 **BRIEF DESCRIPTION OF THE DRAWINGS**

5   FIG. 1 is a schematic diagram of a conventional general-purpose digital computing

6 environment that can be used to implement various aspects of the invention.

7   FIG. 2 shows a conventional tablet and stylus-based computer that can be used to

8 implement various aspects of the invention.

9   FIG. 3 is a flowchart showing simplified steps at a high level of abstraction for

10 implementing a UI architecture according to the principles of this invention upon start up.

11   FIG. 4 is a flow chart showing simplified steps for implementing the "display state"

12 step of the flow chart in FIG. 3.

13   FIG. 5 depicts an example Start Page displayed on a personal viewer according to a

14 preferred embodiment of the invention.

15   FIG. 6 depicts an example book cover page and table of contents displayed on a

16 personal viewer according to a preferred embodiment of the invention.

17   FIG. 7 depicts example Guide Book pages displayed on a personal viewer

18 according to a preferred embodiment of the invention.

19   FIG. 8 depicts example Quick Help pages, one of which is bookmarked, displayed

20 on a personal viewer according to a preferred embodiment of the invention.

21   FIG. 9 depicts an example of a pinned user content page displayed on a personal

22 viewer according to a preferred embodiment of the invention.

23   FIG. 10 depicts an example of inked annotation displayed on a personal viewer

24 according to a preferred embodiment of the invention.

25   FIG. 11 depicts an example End Page displayed on a personal viewer according to

26 a preferred embodiment of the invention.

Introduction

Clutter-Free And Simple

The UI architecture of this invention, also referred to herein as a "documents-with-links UI," supports creation of UI's that have essentially zero clutter, and few concepts to master. In this way, it is a major departure from prior art UI's using a desktop metaphor. The user of a documents-with-links UI according to this invention focuses primarily on content and not on UI appurtenances. Starting with just knowledge of how to page through a document and to follow links, a user can learn how to do any other UI operation. Significantly, the documents-with-links UI works without drop-down menus, toolbars, windows, or other cluttering UI elements (although some of these elements may optionally be made available where they are desired)

1   **Content And UI Are Unified**

2   In the UI architecture of this invention there is essentially no distinction between

3   UI pages and content pages. "UI" and "content" are the same thing, and exist in the same

4   navigation space. As described in more detail below, smart next/previous logic and

5   intelligent management of the navigation chain solve technical problems caused by treating

6   "UI" and content as the same thing in a unified navigational context.

7   Because there is no seam between UI and content, no notion of "dual" spaces, the

8   documents-with-links UI is conceptually simpler for the user than a model that has

9   separate UI and content webs. The one-space model is also more powerful and

10   customizable, as described in more detail below.

11   **Shortcuts**

12   Numerous UI shortcuts and direct manipulations may exist as a configurable layer

13   on top of the documents-with-links UI, so more experienced users can do the most

14   common operations in context, without navigating to UI pages. The user gets the best of

15   terse command access plus the richness of the full browser and answer system for

16   exploring the command set.

17   The documents-with-links UI uses a web architecture, with UI shortcuts layered on

18   top. A naïve user will typically start by using the documents-with-links UI without the

19   shortcuts—that is, by using the Guide Book to access UI functions. An advanced user will

20   typically be able to perform all common operations via the shortcuts and without resorting

21   to the documents-with-links UI Guide Book as often.

22   **Scalability To Various Display Sizes And Types**

23   Various preferred embodiments of the documents-with-links UI will be explained

24   below in the context of a portable "personal viewer" platform. Nevertheless, the

25   documents-with-links UI is scalable across a wide range of device and display types from

26   desktop computers to laptops to hand-held devices. Accordingly, the documents-with-

27   links UI is intended to be implemented on any type of computing platform. The

28   documents-with-links UI exploits a large screen by being able to show content in a book-

29   like way--two full side-by-side pages, as depicted, for instance, in FIG. 8. UI, being

30   content, takes advantage of the large format of pages and the ability to turn/navigate pages

1   (versus the smaller size of menus/dialog boxes in prior art UIs and their limited or

2   nonexistent provisions for navigation).

3   At the other end of the spectrum, the UI scales to small screens because of the

4   flexible selection model that allows command selection regions to be initiated either before

5   or after the desired command is chosen. This means, for example, that on a small screen a

6   user can call up a page of command choices that completely obscure the original document

7   due to screen size limitations, choose the command, then return to the user document and

8   then select the region for the command to operate on. Suppose a user wanted to reformat

9   some text, but had not selected the text to be reformatted yet. The user could press a link

10   that indicates that it will reformat text. The UI of this invention will display an indicator

11   on the screen prompting the user to select the text to be reformatted. After selecting the

12   text, the user can finish the reformatting operation by clicking on another link, such as a

13   link that indicates that the selected text will be reformatted. Alternatively, the user could

14   select the region first and then go find and invoke the command. Being able to select the

15   object upon which a command will operate after selecting the command is unlike the

16   selection model of prior art UIs where object selection must precede command selection.

17   The selection model of the documents-with-links UI is discussed in more detail below.

18   **General Concepts Underlying The Documents-With-Links UI Architecture**

19   **No Distinction Between Content Documents And UI Documents**

20   The basic principle of the documents-with-links UI is that everything the user sees

21   and interacts with is a document. This applies equally to both content and UI. These

22   documents could be implemented in HTML, XML, and the like. There is essentially no

23   difference between content and UI documents. In fact the same document can mix content

24   and UI. Even though some UI elements like context menus and toolbars may be presented

25   in ways that do not look document-like, they are in fact implemented as documents and

26   can be manipulated as such.

27   The uniform treatment of content and "UI" pages is important for several reasons:

28   • Users only need to deal with one set of navigation controls and conventions, and

29   only one navigation space. Users never need to think about whether they are in

30   content space or UI space.

9

1 • Users can use the full power of the UI to manipulate the UI itself. For example,
2   users can search, annotate, customize and edit UI pages the same as any content
3   (subject to permissions). Users can select from multiple views for the page being
4   viewed.

5 • The model naturally accommodates dynamic content, including downloaded
6   content, that mixes content and UI on the same page or as part of a network of
7   related pages.

8 **Links As Commands And Link Properties**

9   A link can be an association between two different spots in a collection of
10 document pages. The spots could be two spots on the same page. The spots could be a
11 spot on one page and a spot on another page. Links can have properties that indicate
12 certain things about the link beyond simply the fact that it relates two different places.
13 What a link relates need not necessarily always be displayable pages. A link can relate a
14 spot in a document and an active runable object such that when a user activates that link or
15 touches that spot in the document, the associated object is run. Links, therefore, can act as
16 commands. Links may be to any kind of command code. Scripts are one example. Binary
17 code objects are another example. As a result, pages that have links replace the drop-down
18 menus and dialog boxes of prior art UIs.

19   A link may manifest in various ways based on the link's properties. Links can look
20 like not only clickable spots, but also fill-in fields and other kinds of well-known and later-
21 developed user interface elements. A document page could have these other kinds of
22 active elements that are really a form of link. Accordingly, activating links goes beyond
23 merely clicking on colored underlined text and can include filling in fields, pressing radio
24 buttons, and the like. Again, pages with links replace prior art dialog boxes having buttons
25 and the like. The links may look like buttons, but the links are simply links.

26   Even the desktop metaphor of files and folders is expressed as lists of links on a
27 page. Additionally, links have properties that govern their appearance and behavior. For
28 example, a link's properties may dictate that it appear as a blue underscored text string as
29 in prior art browsers, or as a 3D button, as a graphic icon, as a thumbnail image of the

1 content being linked to, or even as an embedded frame that is open on the content being

2 linked to.

3   As described in more detail below, a user can modify how a link manifests by

4 modifying the link's properties. The ability to control a link's appearance and behavioral

5 properties makes possible rich authoring and customization of both content and the user

6 interface.

7   Actions in a UI according to the principles of this invention occur by clicking on

8 commands that exist on pages. To the user, a command looks like a link, and in fact it is a

9 link. Specifically, a command is a link whose source anchor is the command hotspot,

10 whose destination anchor is the script or code that implements the command, and whose

11 properties are the command parameters. Some commands may run with canned

12 parameters while others may present users with a form for entering the parameters (for

13 example, the Properties command).

14   Because a command is a link, it has all the functionality of links. For example,

15 commands can visualize in multiple ways, such as an underlined blue label, a button, an

16 icon, or a graphic image. They can be copied from one place to another, moved, and

17 deleted. Their properties can be manipulated, such as to change their appearance or to

18 preset some or all of their parameters. Commands can also be created the same way that

19 any kind of link is created, via a Link command, which establishes a link between a source

20 and destination that the user specifies; in the case of creating a command link, the

21 destination is an executable object such as a command script or binary. Everything users

22 can do to a link, or to objects in general (since a link is an object), users can do to

23 commands.

24   **Guide Book**

25   Users get at commands by navigating to a page where the desired command is

26 found. The documents-with-links UI is organized to make frequently used commands a

27 single navigation step away, or through customizations, no steps away. Less commonly

28 used commands may take more steps to get to.

29   The documents-with-links UI includes a self-explanatory document, the Guide

30 Book. This is literally a readable manual that users can go through in a logical order, a

1 page at a time, like any current user manual. The difference is that each command mention

2 .is an active command instance that can be invoked in place. A variety of Quick Help

3 pages and indices make it easy to get quick access to sets of commands that are commonly

4 used together, that are logically related, or that are typically used as part of a given

5 scenario. Such Quick Help pages could be assembled dynamically based upon the context

6 of the document or documents being viewed by the user.

7 Users can also use search to find commands. This could call into play an

8 intelligent user assistant or other conventional help mechanisms when appropriate.

9 Even if all commands were only one navigational hop away, however, the

10 documents-with-links UI would not be ideal, because users want the most common

11 commands to be zero hops away. Users also want context sensitivity to command

12 presentation, like that provided by conventional context menus. The documents-with-links

13 UI therefore accommodates things like toolbars, context menus, and other UI shortcuts that

14 the user can customize. Like everything else in the documents-with-links UI, shortcuts are

15 implemented as documents with links. The UI shortcuts can be conceptualized as being

16 layered on top of the base documents-with-links UI, yet shortcuts are actually constructed

17 out of the same components that comprise the documents-with-links UI: documents with

18 links.

19 As a user looks up commands from the Guide Book, the user will learn short cuts

20 so that the more a user interacts with the UI, the less often the user will typically need to

21 navigate to the Guide Book. Therefore, shortcuts will be discussed below, with reference

22 to a preferred embodiment of this UI architecture implemented on a personal viewer, as the

23 shortcuts would appear to a user, because, eventually, shortcuts are what most users would

24 use in their day-to-day activities with the documents-with-links UI.

25 **Rich Navigation And Maintaining The User's Navigational History**

26 When a user performs a navigation action, a record is created of where the user

27 navigated to and at what date and time this occurred. It is thus possible to query this set of

28 records to derive many views of a user's navigational history, including, for instance, a

29 network view, also referred to as a history map view. Ways in which a user's navigational

30 history may be viewed are discussed in more detail below in the Viewing History section.

1   The history map view is a generalization of the linear history provided by prior art

2   browsers. The history map view makes it easy for a user to revisit a place the user visited

3   previously, with important cues about the context in terms of other places the user was

4   visiting at the time. The Next command works with the branching history too. A user can

5   explore a chain of links, back up, explore a different chain, back up, and Next the user's

6   way down the original chain to get back to where the user was. This is much easier than

7   having to manually re-follow the original chain of links, which would be impossible if the

8   user has forgotten the chain of links they had previously followed

9   This is an important extension of the prior art browsing metaphor. Unlike prior art

10   browser UIs with their linear navigation chain, the documents-with-links UI doesn't forget

11   all the twists and turns of where the user has been just because the user backs up and

12   proceeds in a different direction. The documents-with-links UI stores not only where the

13   user has been, but also the path(s) the user took to get there. The user can use the history

14   map and/or the Next/Previous commands to get back there again.

15   The Next function works in this network-style navigational context by using

16   heuristics to pick which path forward the user most likely intends. The most basic rule is

17   to pick the forward path along which the user backed to the current node. Other rules

18   provide additional intelligence to account for a user's known navigational patterns, such as

19   whether the user got to the current node by navigating back by individual pages or by

20   groupings of pages (such as by site), or by linking from a parent to a child. The Next

21   function could include options to present a user with a list of forward choices, textually

22   and/or as a map in which the user could zoom into desired areas. In this way, the user can

23   control which branch to take, if the user is not satisfied with the documents-with-links UI's

24   selection.

25   Part of what makes the navigation and context trimming heuristics possible is built-

26   in knowledge of logical levels of information grouping. For example, the documents-with-

27   links UI can include knowledge of collections of pages making up sections and chapters, of

28   collections of sections and chapters making up a document, of collections of documents

29   making up web sites, and so on. The same is true for the layers of command finding and

30   invocation. Such knowledge of semantic clustering helps guide decisions about popping

1   contexts and for presenting the user with reasonable choices about points to jump to along

2   the Next/Previous chain.

3   A problem with treating content and UI as part of the same navigation space is that

4   the user's navigation chain gets cluttered with UI-related pages. The documents-with-links

5   UI solves that through intelligent management of the navigation context, and by making

6   that context a true network, not just a linear chain.

7   Specifically, when a user navigates from one place to another, a new branch in the

8   navigation chain is started. So, if a user's context is currently B in the existing chain of

9   document pages A-B-C, and the user navigates to D, then the new context is D. D might

10   be a UI page the user navigated to from document B. When the user clicks a command on

11   page D, the command executes and removes D from the navigation context. Thus, after

12   finding and executing the command, the user's navigation context is restored to document

13   page B.

14   To find a needed command the user might have to navigate along a chain from D to

15   several other pages in the Guide Book, resulting in a chain of several steps branching off

16   from B. When the user finally picks a command, the documents-with-links UI knows what

17   to act on, and what to remove from the navigation context as follows. Commands operate

18   on the current selection, and, in general, remove the navigation nodes that lie on the branch

19   leading from the current selection to the command. Further, additional heuristics may be

20   used for unusual cases.

21   Current selections in the documents-with-links UI are similar to current selections

22   in prior art desktop UIs, but there are differences because the documents-with-links UI

23   deals with a network of active documents that are different than a desktop of active

24   windows. The two schemes are similar in that every document can have a single, possibly

25   disjoint, selected area. The schemes differ in that the documents-with-links UI can't use

26   the idea of the current focus to decide what selection a command should operate on. In

27   prior art desktop UIs, the document the user wants to operate on typically has the current

28   focus, and all UI elements implicitly reference this focus. In the documents-with-links UI

29   world, the user may have navigated several hops away from the document the user wants

1   to operate on, as the user looked for the desired command, so, in the documents-with-links

2   UI, focus doesn't disambiguate anything.

3   Therefore, in accordance with a preferred embodiment, instead of the current

4   selection being the one whose document has the focus, the current selection is the most

5   recently selected area. All commands that are configured to operate on selections will

6   operate on that most recently selected area. Having executed, they will trim the navigation

7   context at the branch point that leads from the current selection to the command itself.

8   The benefit is that users are free to link into the web of UI pages, exploring them as

9   necessary to find the desired command, and then to invoke it. The act of doing so will end

10   up trimming all the UI navigation from the context, leaving the user back where the user

11   was before navigating to UI pages. Note that depending on the length and content of the

12   navigational path between the command and recent selection, the UI may show the user the

13   target and prompt the user to confirm before proceeding.

14   Note that if no selection exists when a command is executed, then the next

15   selection the user establishes will be considered to be the most recent selection for

16   purposes of determining the command target. In this case, after the user makes the

17   selection, command buttons will be presented in context with the selection by which the

18   user can confirm or cancel execution of the previously selected command. Of course, other

19   suitable methods of determining what selection to operate on are also possible. For

20   instance, one such method is to allow only a single selection to be extant at a time. That is,

21   any time a new selection is initiated, any prior selection is cancelled (un-selected).

22   Another method is to choose what selection to operate on by doing a backwards scan in

23   time order through the navigation context until an active selection is found. In most cases

24   envisioned by the inventors, this latter solution produces the same result as the first one

25   discussed; i.e., the most recent selection will be found.

26   The effect of the algorithm for trimming the navigational context is typically to

27   isolate and remove the branch whose purpose was to find the command that was just

28   executed. Trimming the navigational context of navigation to UI pages does not always

29   occur, however. For example, an Apply command for property setting could leave the

30   property form active and not trim the navigation context. Of course, different commands

1   may choose to operate on the navigation context in different ways. Nevertheless, a couple

2   standard ways will typically cover most of the cases.

3   **Flexible Selection Model**

4   The documents-with-links UI does not require users to make a selection before

5   invoking a command. Users are free to select first and then click a command, or click the

6   command first and then make a selection. If an appropriate selection does not exist when a

7   command is invoked, the user is prompted to make a selection at that time. The selection

8   mechanism and command verbs are designed to give users a lot of latitude about the order

9   in which they do things when carrying out commands. Besides fitting better to users'

10   personal habits, this makes it harder for users to do something "wrong," such as

11   inadvertently applying formatting to text other than the text the user wants to reformat.

12   **Page Display Depends Upon The Link Used To Navigate To The Page**

13   The documents-with-links UI may display the same page of a document differently

14   depending upon which link navigated a user to the page. Suppose the user wants to fill in

15   the TO: field of an e-mail message. In this situation, the user wants to open the address

16   book and make possibly several choices. To accomplish this within the

17   document/navigation metaphor, forms could have special-purpose chooser controls where

18   this provides a good shortcut for the most common choices. But the chooser UI should

19   also make it possible to leverage the full power of the navigation, browsing, viewing, and

20   search UI that is available in list-oriented documents like file folders and the address book.

21   Making a choice from such a list should be a matter of just navigating to that list as a user

22   would in any other context and making the choice.

23   For the address book example, the documents-with-links UI provides a link to the

24   address book that is associated with the input field. The documents-with-links UI makes

25   the link from an input field to its choice document, the address book in this example, be a

26   command with navigation behavior, as opposed to being an ordinary link. This command

27   navigates the user to the document the user needs to choose from and captures anything the

28   user selects. After selecting the addresses the user wants, the user can simply navigate

29   back to the send form where the TO: field is. Alternatively, the user could close the

30   address book or use an OK command, either of which would return the user automatically.

16

1   There is nothing to save, because all choice state is captured as the user proceeds. If there

2   is a change in plan, the user could simply cancel the current selection, or clear the TO:

3   field when the user returns.

4   If the user later wants to change the TO: field choices, the user may click the TO:

5   field link again, and the user is taken back to the address book, with all the current choices

6   still highlighted. The current choices are highlighted because the command that takes the

7   user to the address book picks up the TO: state and paints the required selection regions.

8   To facilitate making selections in scenarios like this address book example, an

9   additional facility is provided. The command which presents the document to choose from

10   can also cause checkboxes to appear next to each entry in the document. Rather than

11   manually selecting items in the document, the user can check or uncheck the checkboxes.

12   This causes the associated entry to be selected or unselected, respectively.

13   In this address book example, the user is accessing a standard document, the

14   address book, and the normal multiple selection idiom to make and change the user's TO:

15   field fill-in choices. The user has the full power of the normal UI for navigating, viewing,

16   and searching, the address book and can navigate to other documents containing addresses

17   where the user can make other choices. Note that because the selection state is associated

18   with the path the user took to reach the address book, the user will see the TO: items

19   highlighted in the address book only if the user links to the address book (or successor

20   nodes) via the TO: field in question. Thus, the existence of an active To: field does not

21   interfere with other uses of the address book or with other active To: fields.

22   This is a significant principle of the documents-with-links UI: the path a user takes

23   to reach a document typically affects the behavior and presentation of the document. This

24   is a way to achieve state-like behavior without requiring special modes or UI mechanisms

25   like dialogs. The implementation of chooser fields is one of the more important uses of

26   this concept.

27   **Modeless UI**

28   Unlike prior art UI's, the documents-with-links UI is essentially modeless. For

29   example, a user could be involved in filling out a form for carrying out a UI operation such

30   as creating a formatting style for a table. In the middle of doing this the user could

17

1 navigate away from this UI form and get engaged in some other UI operation, such as

2 filling out another form for a different purpose, and then, at any time, navigate back to the

3 original, incomplete form. There is no restriction on the number of such incomplete

4 operations that can be in progress simultaneously. Nor is there any limitation on switching

5 away from such incomplete operations. This is unlike the prior art, where complex

6 operations, typically performed via dialogs, must generally be complete or cancelled

7 before the user switches to another activity. Unlike prior UI's, the user of a documents-

8 with-links UI is typically not restricted from interrupting one operation to initiate another,

9 or several others, nor would such an interruption cause the user to lose work already

10 performed in partially completing the first operation.

11 **Creating And Saving Information**

12 Another architectural topic central to the documents-with-links UI is the model for

13 creating and saving information. Any time a user creates something new, the user is

14 creating a new document and linking it into a context. For an object inserted into an

15 existing document, such as an embedded annotation, the user is linking it into the

16 document that is to contain it, with link properties implicitly set to make the object

17 visualize in place (OLE-style embedding). Physically, the object is stored as a child within

18 the parent document's container.

19 For new, standalone objects, like a new word processing document, the object is

20 instead added to the current navigation context, as if the user had done a Next to it from

21 wherever the user was when the user issued the New command. Physically, the object may

22 be stored in the user's sea of "free space" in a hidden system folder, not part of any folder

23 the user is aware of, unless and until the user chooses to file it somewhere.

24 The user does not need to put documents into a filing hierarchy or save them. This

25 is because the documents-with-links UI stores all navigational history. Accordingly, users

26 can find the documents they create by viewing or searching their history map. A user

27 could file a document into a folder as an optional step, using a Save As command or by

28 manually creating a link in the folder that leads to the document (the Save As command

29 could be simply a shortcut for creating such a link). Also, the user could use Save / Save

30 As to update or create versions of a document in the filing hierarchy as desired.

Example Hardware Platforms For Implementing Aspects Of The Invention

FIG. 1 is a schematic diagram of a conventional general-purpose digital computing environment that can be used to implement various aspects of the invention. Computer 100 includes a processing unit 110, a system memory 120, and a system bus 130 that couples various system components including the system memory to the processing unit 110. The system bus 130 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read only memory (ROM) 140 and random access memory (RAM) 150.

A basic input/output system 160 (BIOS), containing the basic routines that help to transfer information between elements within the computer 100, such as during start-up, is stored in ROM 140. Computer 100 also includes a hard disk drive 170 for reading from and writing to a hard disk (not shown), a magnetic disk drive 180 for reading from or writing to a removable magnetic disk 190, and an optical disk drive 191 for reading from or writing to a removable optical disk 192 such as a CD ROM or other optical media. The hard disk drive 170, magnetic disk drive 180, and optical disk drive 191 are connected to the system bus 130 by a hard disk drive interface 192, a magnetic disk drive interface 193, and an optical disk drive interface 194, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer readable instructions, data structures, program modules and other data for the personal computer 100. It will be appreciated by those skilled in the art that other types of computer readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, random access memories (RAMs), read only memories (ROMs), and the like, may also be used in the example operating environment.

A number of program modules can be stored on the hard disk, magnetic disk 190, optical disk 192, ROM 140 or RAM 150, including an operating system 195, one or more application programs 196, other program modules 197, and program data 198. A user can enter commands and information into the computer 100 through input devices such as a keyboard 101 and pointing device 102. Other input devices (not shown) may include a

19

1   microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input

2   devices are often connected to the processing unit 110 through a serial port interface 106

3   that is coupled to the system bus, but may be connected by other interfaces, such as a

4   parallel port, game port or a universal serial bus (USB). A monitor 107 or other type of

5   display device is also connected to the system bus 130 via an interface, such as a video

6   adapter 108. In addition to the monitor, personal computers typically include other

7   peripheral output devices (not shown), such as speakers and printers.

8   The computer 100 can operate in a networked environment using logical

9   connections to one or more remote computers, such as a remote computer 109. Remote

10   computer 109 can be a server, a router, a network PC, a peer device or other common

11   network node, and typically includes many or all of the elements described above relative

12   to computer 100, although only a memory storage device 111 has been illustrated in FIG.

13   1. The logical connections depicted in FIG. 1 include a local area network (LAN) 112 and

14   a wide area network (WAN) 113. Such networking environments are commonplace in

15   offices, enterprise-wide computer networks, intranets and the Internet.

16   When used in a LAN networking environment, the computer 100 is connected to the local

17   network 112 through a network interface or adapter 114. When used in a WAN networking

18   environment, the personal computer 100 typically includes a modem 115 or other means

19   for establishing a communications over the wide area network 113, such as the Internet.

20   The modem 115, which may be internal or external, is connected to the system bus 130 via

21   the serial port interface 106. In a networked environment, program modules depicted

22   relative to the personal computer 100, or portions thereof, may be stored in the remote

23   memory storage device.

24   It will be appreciated that the network connections shown are example and other

25   means of establishing a communications link between the computers can be used. The

26   existence of any of various well-known protocols such as TCP/IP, Ethernet, FTP, HTTP

27   and the like is presumed, and the system can be operated in a client-server configuration to

28   permit a user to retrieve web pages from a web-based server. Any of various conventional

29   web browsers can be used to display and manipulate data on web pages.

1   FIG. 2 shows a tablet and stylus computer that can be used in accordance with

2   various aspects of the present invention. Any or all of the features, subsystems, and

3   functions in the system of FIG. 1 can be included in the computer of FIG. 2. Computer

4   201 includes a large display surface 202 (e.g., a flat panel display) on which a plurality of

5   windows 203 is displayed. Using stylus 204, a user can select, highlight, and write on the

6   display area. Computer 201 interprets marks made using stylus 204 in order to manipulate

7   data, enter text, and execute conventional computer application tasks such as spreadsheets,

8   word processing programs, and the like. One commercially available tablet and stylus

9   computer incorporating many of these features is the Stylistic 2300 computer sold by

10   Fujitsu Personal Systems, Inc., of Santa Clara, California.

11   A stylus could be equipped with buttons or other features to augment its selection

12   capabilities. In one embodiment, a stylus could be implemented as a "pencil" or "pen" in

13   which one end constitutes a writing portion and the other end constitutes an "eraser" end

14   which, when moved across the display, indicates that portions of the display are to be

15   erased. Other types of input devices such as a mouse, trackball, or the like could be used.

16   Additionally, a user's own finger could be used to select or indicate portions of the

17   displayed image on a touch-sensitive or proximity-sensitive display. Consequently, the

18   term "user input device" is intended to have a broad definition and encompasses many

19   variations on well-known input devices.

20   **Example Steps For Implementing Aspects Of The Invention**

21   FIG. 3 is a flowchart showing simplified steps at a high level of abstraction for

22   implementing a UI architecture according to the principles of this invention upon start up.

23   Processing begins at start block 300. In step 302, the system checks to see whether there is

24   a prior existing state. If there is a prior existing state, then that state is used as the current

25   state. If there is no prior existing state, then in step 304 the system retrieves a previously

26   stored default prior state and assigns the value of that default state, which could be, for

27   instance, a default start page, to the current state. In a preferred embodiment, like other

28   content and UI, the current state is stored as a document that contains links.

29   In step 308, the system waits for input from a user. Upon user input, the system

30   determines the nature of the user input in step 310. Gestures for performing UI actions are

21

1   discussed below. The nature of the user input could include a combination of the input

2   device, such as a pen, and the gesture performed by the user. The system then determines

3   the region of the user input in step 312, and, based upon the region of the user input,

4   determines an object associated with the region in step 314. The system then determines

5   an action to be performed based upon both the nature of the user input and the object

6   associated with the region of the input in step 316. The system then performs the action in

7   step 318. The system then loops back to step 308 to wait for more user input.

8   FIG. 4 is a flow chart showing simplified steps for implementing the "display state"

9   step, step 306, in FIG. 3. Referring to FIG. 4, processing begins at block 400. In step 402,

10   the system retrieves a link from the previously saved display state, which may include a

11   list of active or visible links. The retrieved link's display properties are examined in step

12   404. In step 406, a frame is then displayed based on the link's properties. In step 408, a

13   check is performed to see whether, based upon the link's properties, linked-to content or

14   other information about the linked-to document is to be displayed in the link's display

15   frame. If such content or information is to be displayed, then the link's target is examined,

16   as depicted in step 410. Otherwise, there is typically no need to examine the link's target.

17   Then, in step 412, the link (with or without content or information or both from the linked-

18   to document) is displayed in the display frame. Steps 402-412 are repeated until the check

19   for more links in the current display state fails in step 414. Display state processing then

20   ends in step 416.

21   **Preferred Embodiments of the UI Architecture for a Personal Viewer**

22   Various preferred embodiments of the documents-with-links UI of this invention

23   are described herein as implemented on a personal viewer (hereinafter the term "PV" refers

24   to a personal viewer upon which various aspects of preferred embodiments of a

25   documents-with-links UI according to the principles of this invention are implemented).

26   **Personal Viewer UI Elements**

27   Referring now to FIG. 5, an example Start Page is displayed on a PV UI 500 that

28   includes the following elements:

29   •   The display surface 502. This is touch sensitive for both fingertip and pen

30   operations. The touch zones at the four corners 504, 506, 508, and 510 are used

22

to initiate navigation operations such as turning pages and traversing the user's navigational history chain. Labels may be applied to these areas to cue completely naïve users how to turn pages the first time they interact with a PV UI.

The display margins. Documents that present on a PV UI typically show as print-like pages, including margins. The margin space is available for jotting notes and displaying UI-generated features like bookmark tabs and UI shortcuts. A default margin arrangement could be

– Top margin 512: bookmark tabs. Bookmarks or other objects placed here are associated with the containing document.

– Bottom margin 514: document and command shortcuts and toolbar tabs. Document shortcuts or other objects placed here are associated with the display surface.

– Left margin of leftmost display 516: clippings tabs Clippings or other objects placed here are associated with the containing page.

– Right margin of rightmost display 518: clipping tabs. Clippings or other objects placed here are associated with the containing page.

– Inner margin to right of spine 600 (see FIG. 6) (dual display PV UI): no default use.

– Inner margin to left of spine 602: no default use.

In the above list, examples of other objects that may be present in any of the margins are user annotations.

A user can reassign which functions are associated with which margin. Accordingly, the margins may be referred to as the bookmark, document, clip, and command margins, respectively. Of course, other suitable default margin arrangements could also be used. For instance, there could be a zone around the entire document that is outside the document called the edge zone or zones. The edge zones could contain command shortcuts and other kinds of links. Command and shortcuts and toolbar tabs could prefer the lower edge zone, and

links representing the thumbnail view of documents could prefer the left and upper edge zones. Because a page margin would not be needed for carrying command links, clipping tabs could by default use the left margin of the left page for clippings on the left page and the right margin of the right page for clippings on the right page. The inner margins could remain unused, except if the user puts annotations or other links there.

The difference between the margins-only embodiment and the edge-zones embodiment is that, in the margins-only embodiment, the user sees no surrounding frame, and a tap action in the margins has the defaults specified in the list above. In the edge zones embodiment, the user sees a surrounding frame that is wide enough to provide an edge zone, and taps in the corresponding edge zones have the effects specified in the list above. In this latter embodiment, the default action for taps in each of the document's margins is to insert an embedded annotation (i.e., a user note). In the margins-only embodiment, the user would insert annotations into a margin not by tapping there but by explicitly invoking a command, such as an Insert Note command available on the margin's popup menu.

- The display content. This is the area inside the page margins where content is displayed. Content may include links that the user can navigate by touching them, as well as user-created highlighting, jottings, and embedded notes and recordings.

- The note form. This is a document template that is used to implement several important PV UI features, including bookmarks, notes, and clippings. The note form is just a blank document that has predefined fields like an entry field for a title, and option boxes to control the note's presentation and behavior (e.g., its color, type, source back link, and anchor spec). The note form can also have a Send button.

- The built-in documents. The PV has a few built-in documents, as listed below. There is nothing "special" about being a built-in document; these are just

1　documents that a PV UI happens to contain by default. The list can be

2　customized as desired. Possible standard built-ins are:

3　　–　Sign-in Page. This document provides the form for signing in if and when a

4　　　　PV is set in a secure mode.

5　　–　Start Page. An example Start Page is depicted in FIG. 5. This is the central

6　　　　starting point for the PV UI. From here a user can link to any content,

7　　　　including UI, accessible to the user, for instance, documents on the PV itself

8　　　　as well as in the user's desktop workspace and the Internet. The Start Page

9　　　　may take the form of a personal newsletter.

10　　–　My Documents 520. This is a folder listing of all the documents a user has

11　　　　read, authored, subscribed to, or purchased. It is essentially a catalog of a

12　　　　user's personal library. It supports various views to let a user get an

13　　　　overview of what content the user has and to find the content that the user is

14　　　　looking for. My Documents may have links to other PV content like the

15　　　　Address Book and Guide Book, for instance.

16　　–　Guide Book. This is both the PV's help document and the PV's user

17　　　　interface. Example pages of an Guide Book are shown in FIG. 7. The UI is

18　　　　embedded in the Guide Book as links that perform UI functions according

19　　　　to the principles of this invention. The Guide Book includes the Quick

20　　　　Help page of frequently used UI commands. FIG. 8 depicts an example

21　　　　Quick Help page.

22　　–　Map book. A book of maps, including the current site map, history map,

23　　　　topic (browsing) map, net neighborhood, physical vicinity (nearby

24　　　　machines), and local machine.

25　　–　Annotations Folder. This is a folder of the notes, clippings, bookmarks,

26　　　　jottings, and highlights a user has entered into the user's various documents.

27　　　　A user will typically be able to see each of these forms of annotations in the

28　　　　context of the original document to which they relate. These items are

29　　　　stored as external annotations in the Annotations Folder. A user does not

1 usually view this folder, but instead views one of the following persistent

2 views derived from the Annotations Folder.

3 · Clippings Folder. A persistent view of the Annotations Folder, showing

4 only clippings. The default view categorizes by document, with more

5 recently read documents ahead of less recently read documents.

6 · Bookmarks Folder. Same as the Clippings Folder, but showing only

7 bookmarks.

8 · The Notes Folder. Same as the Clippings Folder, but showing only

9 notes.

10 − Notebook. This is a blank document where a user can write notes that are

11 not associated with a particular document. The notebook is provided by

12 default because most users would like to have a notebook to write in.

13 − The mailbox, calendar, and address book. In a preferred embodiment, these

14 will synchronize with a user's desktop or network-based counterparts.

15 **Personal Viewer Display Modes**

16 A PV UI can include a single-display and/or a dual-display. Some dual-display UI

17 actions will cause only the rightmost display image to be replaced. When this happens, the

18 original image may or may not be shifted to the left display when this happens. A rule that

19 can be used is that when the UI action was invoked from a link or menu action initiated on

20 the right display, the image is shifted to the left display and the new page opens on the

21 right. If the action was initiated on the left display, the new page simply opens on the right

22 and no shift occurs. The result is that the new page opens on the right and the page from

23 which it was initiated appears on the left.

24 For a single-display PV, the current display image is simply replaced by the new

25 page. A user can use the Previous and Next functions to flip between the original and new

26 pages.

27 A PV can be held horizontally or vertically. Further, dual-display PV's can treat

28 the two displays as separate pages, or as halves of the same (large) page. The PV can adapt

29 its assignment of case touch zones and display areas to present a consistent geometry to the

1 user regardless of its orientation and display mode. That is, the PV re-maps its definition

2 of things like "upper left corner" and "right margin" to be consistent with its current

3 orientation.

4 **Gestures For Performing PV UI Actions**

5 While other suitable input equipment could also be used, PV UI actions will be

6 described as occurring via touch operations using a fingertip or pen. Users can use a

7 fingertip or the pen, also referred to as a stylus, interchangeably. The pen could have three

8 tips: an erasing tip at one end, a highlighting tip at the other, and, by twisting the pen barrel

9 to extend a small shaft, a writing tip. The fingertip may be recognized as a highlighting

10 tip, with a gesture allowing it to erase.

11 The basic touch idioms are:

12 • Tap (touch and release after a brief time). The action taken depends on where a

13 user tapped. It does not depend on what tip the user is using. The tap would

14 typically last a certain minimum (configurable) time, so that an accidental or

15 glancing touch will be ignored.

16 • Hold (touch and hold contact). The action taken depends on what the user held;

17 usually it opens an object or its context menu. The action does not depend on

18 what tip is being used. The hold time can be a preference parameter.

19 • Swipe. A swipe is any continuous motion of a tip in contact with the display

20 surface. Swiping will write or erase ink or highlighting, depending on which

21 tip is being used. Swipes are also used to make and extend current selections.

22 • Tap-swipe. A tap immediately followed by a swipe starting at the same spot.

23 This causes the pen or fingertip to start a current selection region. Either the

24 writing or highlighting tip can be used. Swiping starting at the current edit

25 cursor location (i.e., the tap is optional when starting on top of the edit cursor)

26 can also draw a selection.

27 • Erase (a jiggling swipe). When done with the fingertip, this is treated as a

28 swipe with the erasing tip. (The motion is the same as erasing with a real pencil

29 eraser). With the pen, a user can just swipe with the erasing tip.

27

1   A user can automatically extend the range of a highlight, current selection, or

2   erasure by holding the pen or fingertip after swiping a part of the range; that is, swipe part

3   of the range, then hold for auto-complete to take over. The range auto-extends to end of

4   word, sentence or line, paragraph, page, section or chapter, and document, in that order, the

5   longer a user holds. The UI provides feedback on the extent of the selection. In tables, the

6   selection auto-extends by cells to the end of a row or column, and then by rows or

7   columns, depending on whether the swipe was across or down. Arbitrary rectangular

8   blocks can be highlighted, selected, or erased by swiping a box shape around the desired

9   area. Selecting the page number of a page selects the whole page.

10   Typically, whatever a user can do with a pen, the user can do with a fingertip,

11   except write ink. While other mapping are of course possible, an example set of

12   keyboard/mouse idioms for the pen idioms and pen tips is:

13   • Tap is left-click

14   • Hold is hold left button

15   • Swipe is hold left or right button and move mouse. Swiping with left button down

16   selects, with right down highlights. There is no need for tap-swipe.

17   • Erase is hold both buttons and move mouse

18   • Jotting is done via the keyboard (i.e., the user gets text instead of ink)

19   • The mouse supports a few other idioms as follows. These and other idioms can be

20   customized.

21   • Right click brings up a context menu, the same one that a hold operation at this

22   location would.

23   • Mouse wheel does page forward and back operations.

24   • Clicking the mouse wheel / third button links to the Quick Help page.

25   **Navigating**

26   **Hyperlinking**

27   One way to navigate is by hyperlinking. The PV UI renders textual hyperlinks

28   using a visual emphasis similar to the way a prior art browser does. Tapping a link will

29   navigate the user to the linked-to place. The user can use the Previous and Next operations

1 to move along the link chain. The PV UI Start Page can exist as the most previous

2 document page in the chain.

3 A user can also hold a link. Doing this will perform a link-specific action. The

4 default behavior for hyperlinks is to present preview information about where the link will

5 take the user; i.e., document name and document position information and/or a thumbnail

6 view of the linked-to document. Continuing to hold the link could cause the preview

7 information to expand into a navigational map of the link context emanating from the link

8 the user is holding. Popup menu choices could also be presented allowing the user to

9 manipulate the link, such as to change its properties.

10 When the user releases a held link no navigation occurs. The user can tap the link

11 to make it navigate. If holding the link opened a navigation map, the map will persist for a

12 short time after the user lets go. This gives the user a chance to tap a spot on the map to go

13 there, or to hold a spot in order to preview and then to expand the map context around that

14 point.

15 **Next And Previous Commands**

16 Another way to navigate is via browser-like Next and Previous commands,

17 performed (in the default assignment) by tapping the PV's lower right and left corners 508

18 and 510, respectively. Specifically, these functions navigate a user along the chain of

19 visitations caused by link operations.

20 Page-turning operations are not part of the next/previous chain. For example, if a

21 user opens a book, reads the first ten pages and then does "previous", the user will return to

22 wherever the user was before the book was opened. The user will not go back to the

23 previous page of the current document (for that, the user can use the previous page

24 operation 504). The distinct page-level and navigation-level functions exist because the

25 page forward/back functions 504 and 506 take the place of the traditional scrollbar for

26 scanning within a single document.

27 **Scrolling Is Possible**

28 Even though the PV UI is page oriented, there are times when a scroll-oriented

29 presentation is the only reasonable display choice. The PV UI supports this by displaying

30 traditional scroll bars when stream-oriented presentation is required. The user may touch

1 the scroll bars to perform the scrolling actions. This looks and works much like

2 conventional scroll bars.

Users Can Create Links

4 PV UI users are not limited to the links that are authored into content; they can

5 create their own. The PV UI implements a rich linking model in which links can have

6 source and destination anchor ranges as well as their own properties. Despite the richness

7 of available features, users can create links as easily as they can copy text, and all with a

8 fingertip; the pen is not required.

Links

Displaying Links

11 How a link is displayed is controlled by its properties. Conventions and heuristics

12 may be used to assign values to these properties at the time the link is created. A simple

13 example is that when the user taps inside the content of a document, a note is created. This

14 entails creating a note document and then creating a link leading from the spot that was

15 tapped to the new note document. In this case the link properties are set to visualize the

16 link as a note icon. As another example if the user highlights some text in a user document

17 and then chooses the "Make link command", followed by the steps to complete that

18 command, the link is made to visualize as a hotspot over the originally selected text, with

19 the selected text changed (for example) to a blue underlined font. Unless a user changes a

20 link's display properties, once these property-setting decisions are made, the link will

21 typically continue to be displayed in the same way.

22 Depending upon a link's properties, when the link is displayed, content of the linked-to

23 document, or other information about the linked-to document, may be displayed in a frame

24 in the linked-from document. For instance, two example previous links, previous link 700

25 in FIG. 7 and previous link 1102 in FIG. 11, depict how examining a link's target allows

26 the PV UI to display information about the destination of a link. For instance, the

27 destination of link 700 is Earth to the Moon, while the destination of link 1102 is the Start

28 Page.

Link Property Sheet

A PV UI link property sheet, like user content and other UI pages, is implemented as a document. It has several pages. The first page is the subset of properties that users would most commonly change, presented in a way that less expert users can understand. Subsequent pages provide the full set of advanced properties and options. Typically, only the most expert user (a content author) would ever modify these.

For example, the first page may contain just the property comments (a notes field), information about where the link leads to, and a chooser that lets a user choose from a few options on how the hotspot should appear (e.g., emphasized text, button, icon, or thumbnail).

The subsequent pages give full access to properties including the source and destination anchor specifications, the visual extent and appearance of the hotspot, and the link behavior options. Each page of the link property sheet is bookmarked so a user can jump directly to it if desired.

Together, the set of link property pages encompasses a lot of information, including general properties, source and destination anchors, hotspot characteristics, and link behavior:

- General properties. These include the type and file system properties of the link. Link type information describes the semantic nature of the link and how it relates the things being linked. Typically, only authoring and viewing software would ever access the type properties, as they pertain to the organization of the material containing the links. Type indicates whether the link expresses a parent, child, or peer relationship, and whether the destination represents:
  - A document component: a figure, table, footnote, or other cross-reference.
  - The next or previous "page" (in the web sense) of the current document/topic.
  - The next or previous document/topic in an authored web of documents/topics.
  - Something not part of the current document/topic: a comment or hyperlink.

- Source anchor. Specifies whether the link is anchored to a character, word, paragraph, image, part of an image, table row, cell, or column, or an arbitrary range of document positions. This parameter is automatically set according to the source selection the user makes when creating the link; this property provides a way to change the anchor.

- Hotspot. Specifies the physical extent and appearance of the hotspot. By default the extent matches that of the anchor, but it can be made bigger or smaller and of arbitrary shape; noncontiguous hotspots are also possible. For hotspots on text, the default appearance is blue underlined text, but the foreground and background colors and text attributes can be changed. Other appearance options include manifesting the link as an icon, button, thumbnail of the link target, or as an in-place (active) rendering of the link target. Hotspots can also be invisible, which is appropriate for links over GIF images, for instance. Another hotspot option specifies how the link is previewed: any or all of: the name of the target, a thumbnail of it, and/or specific balloon text.

- Destination anchor. Specifies the target of the link and its range. The basic link creation UI results in destination anchors that are a single document position (a "point") rather than a range. Destination ranges that are not points are a very advanced feature mainly used in implementing certain viewing and collaboration features. For example, if the destination anchor is a range, the viewing software can automatically synthesize appropriate link preview information based on the content of the range.

- Behavior. Specifies the action to take on tap and hold operations. Choices include
  - Navigate. Goes to the link destination.
  - Preview. Pops up navigational preview information as explained above.
  - Run. Causes the destination content to be executed, with specified run parameters. The target is typically a command or script.

1   Additionally, for links set to appear as thumbnails or in-place renderings, other

2   behavior properties could indicate latency periods for updating the display of

3   the linked-to content relative to a change in the linked-to content by, for

4   instance, specifying a link as hot (display updated often), warm (display

5   updated less often than hot), or cold (display not updated). Further behavior

6   properties could set the pre-fetch, refresh, and caching parameters for hot and

7   warm links.

8

9   **Customizable UI**

10   The user may completely customize the PV UI (unless authored-in content controls

11   prevent editing). Because there is no distinction between the interface and the content, the

12   kinds of normal editing commands the user might use to alter content can be used to

13   customize the interface.

14   **Users Can Place Links In UI**

15   For instance, links are not restricted to existing only in the content area of the

16   display. They can exist anywhere. A user could, for example, place a link over a

17   bookmark; the link would take precedence for tap operations, meaning that the bookmark

18   would act as a document-associated shortcut to some other document; whereas a bookmark

19   is normally a link within the current document.

20   **Page Pinning**

21   The PV UI display may be divided into regions in which different content may be

22   displayed. Unlike prior art UIs, each region may be individually navigated without

23   changing focus or "window" state. Each region may include concurrently active links that

24   may be activated without changing focus. This aspect of the current invention is referred

25   to as page pinning. Page pinning provides access to and interaction with multiple sources

26   of content concurrently, while reducing the complexity associated with both the design and

27   use of the interface.

28   In prior art user interfaces, each window runs a separate application that has state

29   information associated with it. For instance the state information for a word processor

30   might be that it is currently in insert mode, or replace mode and what the current font is.

1   The user often has to keep in mind which application is running in which window and

2   what the state of that application is to avoid unexpected results.

3   In the PV UI, unlike prior art user interfaces, there is no notion of a selected one of

4   multiple windows having the current "focus." Any display region may include active

5   links. The user can touch anywhere on the screen and will get an appropriate action based

6   on what the user touches. For instance, if a user has a page pinned as depicted in FIG. 9,

7   and both pages being displayed contain links the user can simply touch a link in either

8   page and the link will be activated.

9   Superficially, page pinning resembles a multi-windowing user interface. The same

10 may be said for dialog boxes that may be included in a documents-with-links UI according

11 to the principles of this invention. Nevertheless, unlike this invention, multi-windowing

12 UIs contain more than one UI. Each window in a multi-windowing UI gets its own UI,

13 with its own state information, such as history information, and its own UI features, such

14 as menu or tool bars. According to the principles of this invention, pinned pages, like all

15 other content (the UI included), share a single UI having only one set of state information

16 and having a unified navigational history. There are no separate UI controls, such as menu

17 bars or tool bars, for the separate concurrently displayed pages. The unified navigational

18 history is discussed in more detail below.

19   When the user has pinned a page and is viewing two pages side-by-side, the user

20 essentially has two different view ports into a single navigation context. A set of

21 documents exists in a navigation context. The navigation context is essentially a record of

22 every place a user has ever navigated to and when the user navigated there. The navigation

23 context can be thought of as a map or history of the user's navigation. A visible frame,

24 such as a pinned page is just a view port into a single shared history. The only state

25 associated with a displayed page-pinning frame is an indication of the document page that

26 is currently being displayed.

27   Page pinning is described further in commonly assigned U.S. Patent Application

28 Serial No. _____, entitled Methods And Apparatus For Displaying Multiple

29 Contexts In Electronic Documents, filed contemporaneously herewith, which is

30 incorporated herein by reference.

1 ### Navigating Documents/Link Properties

2 The PV UI contains methods for paging within a document (previous/next) and for
3 traversing chronology (history through back/forward). Attaching properties to links
4 enables a rich model for navigation. For example, holding on a link offers additional
5 information about where that link will lead. Link preview information could be displayed
6 in thumbnail form and further holding could result in a graphical map of the links attached
7 to that prospective destination. Back and forward buttons, by default, display thumbnail
8 views of the pages that tapping that button will lead to.

9 ### Viewing History

10 History (which could include all the pages the user has viewed) may be viewed in a
11 number of ways: by time, by appearance, by site, document, section, page, and the like.
12 Since a user's navigation history is saved, the users' sequence may be viewed: as nodes
13 with side tracking branches, as a linear list, or as a combination of most recently viewed
14 pages and last few task categories. For example, a query could perform a multilevel
15 categorization by document id resulting in a hierarchy that represents all the forward
16 navigation paths a user has taken from any given document. By restricting (filtering) this
17 categorization to a particular time frame, the navigation network as it existed at a particular
18 point in time can be shown. Of course, other kinds of views are also possible. For
19 example, the history can be categorized by higher level groupings like web sites or user
20 tasks, with the results presented in alphabetical rather than time order, thus letting users
21 return to a previous location according to the kind of information or activity, as opposed to
22 the time during which, the location was last visited. To make certain views such as the
23 network view more efficient to recreate, additional property information may be stored on
24 each navigation record. Accordingly, maintaining the navigation context as a database of
25 navigation records allows support for a rich variety of queries and views.

26 ### Applies To UI

27 Because the UI is built out of content, features used to search text, change viewing
28 options on lists and tables, annotate, and the like all work for UI as well as for user content.
29 As an example, the contents of any view, such as a view of a filing or command hierarchy,
30 become searchable, sortable, and annotatable.

1 <u>Non-Linear Navigation History</u>

2   A key difference between the navigation mechanism of the PV UI and prior art

3 browsers is that the PV UI, unlike prior art browsers, maintains a non-linear navigation

4 history. For example, if a user navigates through a set of links then goes "back" several

5 steps, and then navigates through a different set of links, the user is still able to go "back"

6 and then retrace the original path of links they navigated. With prior art browser's this is

7 not possible—recollection of the first set of documents that were "backed" over is lost.

8   Further, the PV UI maintains a nonlinear navigation context that records every

9 place a user has ever navigated to, when in time the user was there, and where the user

10 went to from there. This, in turn, allows a user to navigate from general content pages into

11 user interface pages, perform interface functionality, and then return to the user's

12 documents without losing what the user's navigational context was before navigating to

13 the user interface pages. As described in more detail above in the Rich Navigation And

14 Maintaining The User's Navigational History section, this invention automatically

15 removes navigation from the beginning of navigation into UI pages that lead up to

16 performance of some UI functionality so the user's navigational history does not get

17 cluttered with navigation within UI pages. Saving the user's navigation history may also

18 help a user retrace navigational steps that the user would not otherwise be able to

19 remember. For instance, suppose a user does not remember where they were when they

20 viewed some content they liked. If the user remembers where they had been before they

21 viewed that content, then the user can navigate to this prior place and query navigational

22 history for everywhere they had navigated to from that particular location.

23   This is very unlike prior art multi-windowing UIs in which each application has its

24 own navigation history that can not be integrated with the history of other applications a

25 user is concurrently running. For example, suppose a user runs a word processor and a

26 browser concurrently on a desktop computer. The user can switch between them in a way

27 well known in the art. The word processor and the web browser will both have their own

28 state information. In other words, the word processor and the web browser will each be

29 separately nested in their own experience of history or navigation. As a result, if the user

30 switches from the word processor to the browser, visits a few web sites and wants to return

1 to the word processor, the user will not be able to get back to the word processor by hitting

2 the browser's "back" button. The browser and the word processor essentially exist in

3 separate contexts, with each context having its own state information and unique history.

4 **Annotations**

5 **Making Annotations**

6 The PV UI allows a user to interact with content and the UI to make either or both

7 more memorable, via bookmarks, clippings, highlights, overlaid and embedded ink and

8 audio notes. Bookmarks are described further in commonly assigned U.S. Patent

9 Application Serial No. _____, entitled Bookmarking and Placemarking a

10 Displayed Document in a Computer System, filed contemporaneously herewith, which is

11 incorporated herein by reference. Example bookmarks 800 and 802 are depicted in FIG. 8.

12 Ink annotations are described further in commonly assigned U.S. Patent Application Serial

13 No. _____ entitled System And Method For Annotating An Electronic Document

14 Independently Of Its Content, filed contemporaneously herewith, which is incorporated

15 herein by reference. Audio annotations are described further in commonly assigned U.S.

16 Patent Application Serial No. _____, entitled Annotating Electronic Information

17 with Audio Clips in an Electronic Device, filed contemporaneously herewith, which is

18 incorporated herein by reference. These annotations may be performed with not only

19 different input methods, but adapted to the most convenient or natural one, such as a finger

20 for highlighting, a stylus for writing or doodling, and speech for lengthier commentary.

21 **Storing Annotations**

22 The annotations exist apart from the content (as files with links to the content);

23 therefore annotations can be displayed not only layered on the content in appropriate

24 positions within the content, such as, for instance, note 1000 in FIG. 10, but in other

25 locations and visualizations. That is, each of these annotations is available to the user not

26 only *in situ,* in the content where it was created, but cross-referenced in automatically-

27 created indices which can be manipulated much as described in the Viewing History

28 section above: by time, by appearance, by site, document, section, page, etc. Likewise,

29 annotations could be shared, selectively shared, or kept private with the kind of

30 functionality described in the Collaboration section below.

Interface Shortcuts and Smarts

Most-Likely-To-Use-Links and Other Heuristics

The PV UI can include heuristics, which allow it to offer up most-likely-to-use links to additional material. One case of these is the context menu that appears when the user holds the Next button; in addition to the most recent documents the user has visited subsequent to the current one, the Next menu includes heuristically determined choices of other places the user may wish to visit (for example, documents on a topic related to the current one). Another case is a feature called an End Page, such as, for instance, End Page 1100 depicted in FIG. 11. Such an End Page is essentially a summary page at the end of a document or book that offers the user related topics such as "additional works by this author/on this topic/written in the same time period/commented on by these critics" etc. In the case of directories such as email, where each message is technically a document, the end page offers threads based on that message/other messages by same author/other text on the same topic, and the like.

Of course, other heuristics could also be used depending on the context. For example, when offering up a set of command choices to a user, the heuristics could be based on analysis of the user's current document context, recently used commands, and commands the user has used most frequently in this and similar contexts in the past. Such heuristics are described further, in the context of navigating to a particular portion of the Guide Book, in commonly assigned U.S. Patent Application Serial No. _____, entitled Method and Apparatus for Providing Help and Settings Control to Users of an Electronic Book, filed contemporaneously herewith, which is incorporated herein by reference.

Frequently Used Links

The PV UI could also provide commands (actually links) embedded in the page that are related to the kind of material on the page. In a mail message for example, "reply" "reply all," and "forward" are links so frequently used as to warrant their inclusion directly on the page. There may be other commands or links used so often that they are dynamically bubbled up, or in the case of linear reading, used so infrequently that none

1   appear. The second level of visibility—"hold on the page for more information"—offers a

2   more extensive list. If none of these satisfy the user's need, the user can summon the

3   Guide Book.

4   **Start Page**

5   The top level of the PV UI provides the user with a Personal Newsletter or Start

6   Page, which is the launch point for many activities. An example Start Page is depicted in

7   FIG. 5. Highlights might include urgent mail messages, projects, PIM items, documents,

8   news, which could be divided into system-offered default components, such as Messages,

9   Notes, Clippings, and the like. Favorite links that the user wants to keep readily available

10   could also be included. What appears on the Start Page can be both user-configured as

11   well as configured from profile information the PV UI has learned by observing a user's

12   browsing and e-mail reading patterns and the like.

13   **User Can Modify Start Page**

14   Advanced users can perform the same kinds of operations on the interface itself as

15   those performed on content. If the default components on their Start Page are not to their

16   liking, they can delete or alter even system-offered components such as Mail Messages.

17   Or, for instance, if the automatic page number used by the bookmark header is insufficient

18   for recognition, they can alter its text, color or any other property.

19   **Audio Recording and Annotation**

20   The PV UI supports the creation of audio clips that can be used for annotation of

21   any displayed document. The clips are based on a timeline model in which audio (or

22   video) recording is a data stream parallel to and synchronized with the material in a

23   document. If the user has changed pages while recording, then the clip when played back

24   will also change the page when the appropriate place in the audio clip is reached. The

25   interface supports both document and page-specific audio notes as well as global

26   recording. The interface is presented with cassette-like controls, including index

27   forward/back and editing. Each clip is stored as an individual document and can be sorted

28   and filtered to present the clips in multiple ways (all audio on a given page, all audio for a

29   given book, audio notes in the order they were recorded, and the like). Additionally, the

30   audio clip recorder supports intelligent VOX for hands-off note taking. A further feature is

39

1   the ability to embed audio notes at specific points in the content of a document in a manner

2   similar to creating footnotes. These audio notes can be created and played back by a single

3   tap on the screen at the point where the audio note was (or is to be) inserted. The presence

4   of embedded audio notes is signified by a small icon in the content that is laid out in a

5   manner similar to a footnote symbol. As previously mentioned in the Making Annotations

6   section, audio annotations are described further in commonly assigned U.S. Patent

7   Application Serial No. _____, entitled Annotating Electronic Information with

8   Audio Clips in an Electronic Device, filed contemporaneously herewith, which is

9   incorporated herein by reference.

10

11   **Guide Book**

12   Because the commands in the PV UI are simply links on a document page, there is

13   no difference between documentation, help, wizards and the other ways the PV UI helps a

14   user perform an action. Asking for help by clicking on a help link summons an appropriate

15   help page composed on the fly from the content of the Guide Book. The principles that

16   underlie this composition are contextual: that is, if the user is currently in the "list" or

17   "books" of Mail Messages, the help system infers that requests for assistance are about

18   how to perform work with Mail Messages.

19   For instance, FIG. 8 shows two pages of an example Guide Book. The page on the

20   right, page 9 of 10 includes descriptions of commands, such as the "PIN THIS PAGE"

21   command 800. The Guide Book describes what the command does. For instance, for

22   "PIN THIS PAGE" the description states that "Tapping on this command will "freeze" a

23   secondary copy of the current page and display it as a floating page over the book." FIG. 9

24   shows an example of a pinned page. In addition to describing this command, the text "PIN

25   THIS PAGE" is a link that, when activated, will execute the "PIN THIS PAGE" command

26   on either the most recently selected page or on a page to be selected after activating the

27   "PIN THIS PAGE" command link.

28   **Obtain Help And Perform Functions In The Same Way**

29   Unlike prior art UIs, the PV UI is constructed as a set of document pages just like a

30   user document or a web site is constructed as set of document pages. This invention puts

1   all of the user's content pages and user interface pages into a single grouping of

2   information that the user can access. This invention then leverages well-known browser-

3   like navigational capabilities to allow a user to navigate back and forth between various

4   pages and to put links to pages in favored lists and the like. All of this is leveraged to

5   provide the means by which a user navigates to user interface pages.

6   The PV UI can provide context-specific dynamically synthesized views of links to

7   give the user direct access to relevant UI help pages. The user can search for user interface

8   functionality in the same way the user can search other document pages. Of course, pop-

9   up menus and other devices such as permanent links placed on the screen that give the user

10   direct access to UI help information can also be provided.

11   Because the Guide Book is simply content, operations that may be performed on

12   other types of content, such as the Start Page, the UI, and content in general may also be

13   performed on the Guide Book. Unlike prior art user interfaces, the way a user gets help for

14   performing a function and the way the user performs the function are the same. To

15   perform a function and to get help for a function, the user simply activates a link.

16 **Concluding Remarks**

17   The foregoing has described a user interface architecture based on documents-with-

18   links that facilitates creation of user interfaces that allow users to read, annotate,

19   collaborate and perform other tasks typical of knowledge work, as well as alter the

20   interface to best suit their work patterns. It will be appreciated that many modifications

21   and variations of the invention are possible, and the specific examples and descriptions

22   herein do not limit the scope of the invention.